UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/343,805	06/30/1999	RONNIE I. CHAIKEN	777.285US1	8333

27488 7590 04/18/2003

MERCHANT & GOULD
P.O. BOX 2903
MINNEAPOLIS, MN 55402-0903

EXAMINER
STEELMAN, MARY J

ART UNIT	PAPER NUMBER
2122	18

DATE MAILED: 04/18/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/343,805	CHAIKEN ET AL.	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 March 2003* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15 and 17-43* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15 and 17-43* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

RCE - DETAILED ACTION

1. Claims 1-15 and 17-43 are pending.

2. As per Applicant's request, claim 15 is amended, claim 16 was previously cancelled without prejudice.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefore, subject to the conditions and requirements of this title.

4. Claims 19 – 30 are rejected under 35 USC because the claimed inventions are directed to non-statutory subject matter.

As to claims 19 – 24, merely claimed as "A computer-readable medium" that merely stores a data structure comprising various fields, flags, and operands without creating any functional interrelationship, either as part of the stored data or as part of the computing processes performed by the computer ("acts"), then such descriptive material alone does not impart functionality either to the data as so structured or to the computer. Thus, such "descriptive material", non-functional descriptive material, that cannot exhibit any functional interrelationship with the way in which computing processes are performed does not constitute a statutory process, machine, manufacture or composition of matter. The purely non-functional descriptive material cannot alone provide the practical application for the manufacture. **Warmerdam**, 33 F.3d at 1361, 31 USPQ2d at 1760. **In re Sarkar**, 588 F.2d 1330, 1333, 200 USPQ 132, 137 (CCPA 1978). *See* Examination Guidelines for Computer-Related Inventions-Final Version, page 10. *See* M.P.E.P. PP 2106(IV)(B)(1)(b).

As to claims 25 – 30, merely claimed as "A computer-readable medium" that merely stores a data structure comprising various fields and pointers to other data structures without creating any functional interrelationship, either as part of the stored data or as part of the computing processes performed by the computer ("acts"), then such descriptive material alone does not impart functionality either to the data as so structured or to the computer. Thus, such "descriptive material", non-functional descriptive material, that cannot exhibit any functional interrelationship with the way in which computing processes are performed does not constitute a statutory process, machine, manufacture or composition of matter. The purely non-functional descriptive material cannot alone provide the practical application for the manufacture. **Warmerdam**, 33 F.3d at 1361, 31 USPQ2d at 1760. **In re Sarkar**, 588 F.2d 1330, 1333, 200 USPQ 132, 137 (CCPA 1978). *See* Examination Guidelines for Computer-Related Inventions-Final Version, page 10. *See* M.P.E.P. PP 2106(IV)(B)(1)(b).

Claims 19 – 24 and 25 – 30 merely define a template for data structures and their contents. Because they are non-functional (without functional interrelationship), the data structure merely holds names, not the functions themselves, they do not comprise of any computing processing activity and are thus considered non-statutory subject matter.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1, 2, 6 - 9, 11 – 13, 15 - 18, 31 – 35, and 39 – 41 are rejected under 35 U.S.C. 103(a) as being anticipated by Breslau et al., U.S. Patent 6,345,311, in view of Srivastava, U.S. Patent 5,966,539.

Breslau disclosed a method and system related to objects in a heterogeneous execution environment. An object identified as being needed in a second environment saves state, and through the use of data format tables (DFT) and system configuration tables (SCT) the object is instantiated in another environment, preserving state. Although Breslau's invention does address a heterogeneous program, having components in different forms, that are transformed, without resorting to the original source code, Breslau does not disclose an intermediate language step in the translation. However Srivastava did disclose object code modules translated into an intermediate language that is compatible with a plurality of computer system hardware architectures. (Abstract, lines 5-8.) Modified code is converted into executable code compatible with a target one of said plurality of computer system hardware architectures. While Applicant places emphasis on the "target one" phrase, Examiner interprets this to mean that executable code could also be derived for a second, third, etc. of said plurality of computer system hardware architecture, thus heterogeneous.

As to claim 1:

Srivastava539 discloses a computer system that translates and transforms a plurality of source code modules (Examiner considers modules to be equivalent to components.) into an intermediate representation, using a symbol table to resolve addressing issues, partitioning blocks and procedures, analyzing the flow within and between program procedures, optimizing and producing platform specific executable code (See abstract on page 1.) and comprising:

"...*obtaining a binary for a component*..." (See fig. 3.) Binaries are input as executable code.

"...*determining a plurality of basic blocks*..." (See fig. 3.) Basic blocks are displayed at 101 – 103. Srivastava539 discloses (col. 4, line 67 – col. 5, line 1), "...procedures are organized into basic execution blocks..."

"...*translating each...instruction...into an intermediate representation instruction*..." (See fig. 3.) At step 52, code is translated into "register translation language" which as stated in col. 2, lines 53 – 55, "The register transfer language is independent of any particular computer system hardware architectures." The Examiner considers "register transfer language" to be equivalent to "intermediate representation."

"...*determining a procedure*..." In Fig. 3 and Col. 4, lines 63 – 64 Srivastava539 discloses, "The organizer 54 partitions the linked code 52 into a collection of procedures 100."

"...*creating an intermediate representation of the procedure*..." Srivastava539 also discloses (col. 5, lines 7 – 10), "...procedure flow graphs (PFG)...the PFG maps the flow of control through the basic blocks..." Examiner interprets this as mapping the flow of instructions through the blocks comprising a procedure.

"...*annotating the intermediate representation...with symbol information*..." Srivastava539 also discloses (fig. 3 and col. 5, lines 39 – 41) "The intermediate representation of the program is in the form of the register transfer language and the logical symbol table 51"

"*creating an intermediate representation of the component*..." Srivastava539 also discloses (col. 5, lines 7 – 12), "The organizer builds...a program call graph (PCG)...The PCG indicates how the procedures are called by each other." Also, col. 11, lines 60 – 61 state, "FIG. 8

shows a program having procedures 320-329 calling each other as defined by a program call graph..." Examiner considers groups of procedures calling each other to be equivalent to components.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to have modified Breslau's invention to transforms objects in a heterogeneous environment by including the features disclosed in Srivastava to translate objects into a neutral language because both are transforming objects, whereas Srivastava is providing more details regarding the steps in the transformation.

Per claim 2:

"...creating an intermediate representation of the heterogeneous program from the intermediate representation of the component." Srivastava539 also discloses (col. 4, lines15 – 16), "A linker 50 embodying the invention can combine the object modules..." See the linker (50) in figs. 2 and 3. Note that the linker processes the intermediate representation.

Per claim 6:

"...*symbol table...to define an address space*" Srivastava539 also discloses, (col. 5, lines 21 – 22) "All symbolic addresses are resolved..." Figure 3, item 53 shows the logical symbol table, LST.

"...*Translating...into a platform-specific*" Srivastava539 also discloses, (col. 5, lines 20 – 21) "The code generator 57 generates machine-dependent code for the target hardware architecture." Examiner considers platform-specific to be equivalent to machine-dependent code. Fig. 3, item 9, CPU Architecture Description is a collection of platform specific information.

Per claim 7:

"...*Outputting the symbol information*" Srivastava also discloses in Fig. 3 the LST outputting information to the organizer, item 54.

Per claim 8:

"...*Outputting emitted block information*" Srivastava also discloses in fig. 3, item 100, comprised of blocks, item 105, with an output to the code generator, item 57.

Per claim 9:

"...obtaining the emitted block information..." See fig. 3, item 100 for stored collection of blocks, 105 in the form of intermediate representation.

Per claim 11:

"...*replacing ...platform-specific instruction with a platform-neutral...*" Srivastava also discloses, (fig. 3 and col. 4, lines 41 – 48) "During an initial phase, the translator transforms the collection...The linked code module is in the form of an intermediate register transfer language..."

"...*replicating a complex platform-specific instruction in an intermediate representation...*" Srivastava539 also discloses, (Abstract, line 1) "A computer system is directed to convert a program written as a plurality of high level source code modules..." Examiner considers a plurality of high-level source code to be equivalent to complex platform-specific. See above for the treatment of converting platform specific to an intermediate representation.

Per claim 12:

"...intermediate representation of the program arranged in a hierarchy..." See fig. 3. Blocks are displayed as item 105, procedures are displayed as items 101 – 103, components, 100,

are created from the procedures and sent to the code generator. Instructions that have been processed by the translator are in the intermediate representation form.

Per claim 13:

"...code block element references...instruction for multiple instances of a platform-specific instruction..." See fig. 3. A code block as shown in item 100 references intermediate representation instructions for multiple instances of platform specific instruction. Later in the processing, the CPU Architecture Description, item 9, works with the Code Generator, item 57, to make code platform-specific.

Per claim 15:

"...reading a heterogeneous program having a plurality of executable components in at least two different forms; obtaining a binary for a component contained in the heterogeneous program; analyzing the binary; building an intermediate representation for the component; and building an intermediate representation of the heterogeneous program containing the component."

Breslau: Abstract, lines 1-7 and figure 10. Once an object has been instantiated, it can be transformed to another heterogeneous environment through the use of transformation tables.

Srivastava: See response to claim 1 above. Also see figs. 2 and 3 where code is input, analyzed and built into an intermediate representation.

Per claim 17:

"...analyzing the binary...performing code discovery...determine a plurality of basic blocks...establishing block relationships..." See response above as disclosed by Srivastava539. Also see fig. 4 and col. 9, lines 37 – 39 in which Srivastava539 discloses, "The basic blocks 151-

155 of the procedures 101-103 of the program 20 are repeatedly examined during successive passes..."

Per claim 18:

"...building the intermediate representation ...comprises translating every instruction...into an intermediate representation..." See response above. Fig. 2 shows the optimizing linker, which is further subdivided in fig. 3, items 51,52,53,54. In fig. 3, translated instructions are held in the RTL Program Module, item 52.

Per claim 31:

"A computerized system comprising: a processing unit; a system memory coupled to the processing unit through a system bus; a computer-readable medium coupled to the processing unit through a system bus; a translation and transformation system executed from the computer-readable medium by the processing unit, wherein translation and transformation system causes the processing unit to translate a platform-specific binary corresponding to a component in a heterogeneous program, into a plurality of intermediate representation instructions." This is a version of the claimed method discussed above, in claim 1, wherein all claim limitations also have been disclosed and/or covered under the same cited areas as set forth above. Moreover, see Figs. 1-3. Thus, the same rationales provided in the rejection of claim 1 above is also applied and incorporated herein.

Per claim 32:

"Application program interface instructs the...system to further cause the processing unit to transform the plurality of intermediate representation instructions." See figs. 1, 2 and 3. The analyzer further transforms the instructions as shown in figs. 4 and 5. Srivastava539 also

discloses, (col. 13, lines 56 – 58) "The code generator, using the CPU description can produce machine executable code from the rearranged linked module." Srivastava539's system is directed by a program (API) to process input by translation and transformation on a plurality of intermediate representation instructions. Fig. 9 shows additional optimizing transformations.

Per claim 33:

"*...translation and transformation system further causes processing unit to translate...into a modified platform-specific binary.*" Srivastava539 also addressed and /r covered such claimed limitations as set forth in claim 32 above.

Per claim 34:

"*...processing unit to translate the modified platform-specific binary into a modified plurality of...instructions...*" See fig. 3, items 9, 57 and 60. Srivastava539's system translated the modified platform-specific binaries. A plurality of modified instructions is produced for further transformation.

Per claim 35:

"*...translating a platform-specific binary corresponding to a component in the heterogeneous program, into a plurality of intermediate representation instructions.*" This is an apparatus version of the claimed system discussed above, in claim 34, wherein all claim limitations also have been disclosed and/or covered under the same cited areas as set forth above. Moreover, see Fig. 3, item 9, which shows a collection of platform-specific descriptions used in the translation. Thus, the same rationale provided in the rejection of claim 34 above is also applied and incorporated herein.

Per claim 39:

"...translating ...instructions into a new version of the platform specific binary."

Srivastava539 discloses the new platform specific binaries in figs. 2 and 3, item 60. Col. 4, lines 21 – 22 state, "The linker produces machine dependent executable code 60 which can be loaded in the memory 3 for subsequent execution by the CPU 2 of the computer system 1 of FIG.1"

Per claim 40:

"*...translating the new version of the platform-specific binary into a new version of the plurality of intermediate representation instructions.*" See fig. 3. By supplying the new version of platform-specific binary into the system at item 60, a new version of the plurality of intermediate representation instructions can be processed.

Per claim 41:

"...iterating an intermediate representation of a heterogeneous program...create a plurality of new versions..." This is an apparatus version of the claimed methods discussed above, wherein the claim limitations also have been disclosed and/or covered under the same cited areas as set forth above. Moreover, see Figs. 1-3 where multiple executions through the system can be used to create a plurality of new versions of the heterogeneous program. Thus, the same rationales provided in the rejections of the above claims is also applied and incorporated herein.

7. Claims 3 – 5, 10, 36 – 38, 42, 43 are rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau et al., U.S. Patent 6,345,311, in view of Srivastava, U.S. Patent 5,966,539, as applied to claims 1, 6, 35 and 41, respectively above, and further in view of Srivastava, U.S. Patent 5,539,907.

As to claims 3, 10 and 36, as applied to claims 1, 6, and 35 respectively above:

Breslau disclosed transforming objects in a heterogeneous environment, Srivastava539 disclosed the claimed invention, adding transformation details regarding intermediate representation of code. He did not explicitly disclose the concept of "*user input*" to affect control of the transformation and translation process. However, Srivastava907 discloses a translation program that includes instrumentation and monitoring by factoring in user input parameters. Srivastava907 (col. 6, lines 44 – 47) teaches, "Under the user's direction, portions of the program to be monitored are identified by "navigating" through the program using the program and CPU description and the graphs." Also, (col. 6, lines 62 - 64) teach, "The user supplied UIR [user instrumentation routines]...locate and modify specific portions ..."

It would have been obvious to one skilled in the art, at the time the invention was made, to modify the teachings of Breslau and Srivastava907, by providing methods for user input, into the teachings of Srivastava539 (one or more iterations of translation and transformation of platform-specific heterogeneous programs, prior to creating new versions of the heterogeneous programs), in order to provide a more useful customized translation system, because user inputs can be used to obtain precise information reflecting the operation of the computer system.

As to claim 4, as applied to claim 3 above:

Srivastava539 does not teach the method of "instrumenting". However, Srivastava907 teaches "*instrumenting*" as a means of monitoring control flow. Srivastava907 discloses, (Abstract, lined 10 – 12) "Fundamental instrumentation routines identify, locate, and modify specific program components to be monitored." Thus, is would have been obvious to one skilled

in the art, at the time the invention was made, to modify the teachings of Srivastava907, by including instrumentation routines, into Srivastava539 (methods of transforming intermediate representations of the program during the translation and transformation of heterogeneous programs), because by identifying and locating specific program components, the components can be modified to call user analysis routines, which may help to better optimize performance.

As to claim 5, as applied to claim 3:

Srivastava539 does not teach optimizing the intermediate representation of a program based on the "*user input*" limitation (which is addressed above in claim 3). However, Srivastava907 also teaches "*...optimizing the intermediate representation of the program...*" (See col. 3, lines 49 – 52), "Performance data can be used to optimize the design..." Also, (col. 3, lines 63 – 64) "It is desirable to monitor the execution of the program at the procedure, basic block, and instruction level." Also, (col. 14, lines 13 – 16) "The common structures of programs to be identified and modified can be performed by fundamental instrumentation procedures." Thus, it would have been obvious to one skilled in the art, at the time the invention was made, to modify the teachings of Srivastava907, by optimizing the intermediate representation of a program based on user inputs, into Srivastva539 (methods of transforming intermediate representations of the program during the translation and transformation of heterogeneous programs), because optimization produces a faster, more efficient code, which will increase the performance of the program.

As to claim 37, as applied to claim 36 above:

"*...translating the plurality of intermediate representation instructions...into a modified platform-specific binary*" Srivastava539 also teaches and/or addresses the translation

intermediate representation instructions into platform-specific binaries. See fig. 3. Intermediate representation instructions are held in item 52. From that point they are further processed into platform-specific binaries, using the CPU Architecture Description, item 19, as per the diagram. User input, item 9a, is a factor in the system. Thus, it would have been obvious to one skilled in the art, at the time the invention was made, to also implement the teachings of Srivastava539, to translate the plurality of intermediate representation instructions, into the combination of Srivastava539 and Srivastava907 (in accordance with, input parameters, as set forth in claim 36 above), because translating instructions into binaries that have been customized with user inputs can produce more specialized efficient program code, which will run faster on a specific computing system.

As per claim 38, as applied to claim 37 above:

"*...translating the modified platform-specific binary into a modified plurality of intermediate representation instructions for further transactions.*" Srivastava539 also discloses a method for translating the modified platform-specific binaries (See the Srivastava539 figs. 2 and 3.), but does not explicitly address doing this in accordance with user inputs as in claim 36. However, Srivastava907, of the combination of Srivastava539 and Srivastava907, also disclose "*user inputs*", as set forth in claim 36 above. Thus, accordingly such claimed limitations would also have been obvious.

As per claims 42 and 43, as applied to claim 41:

"*...manipulating the intermediate representation using data input...*" and "*terminating the iterating...based on data input...*" Srivastava539 disclosed the claimed invention, as discussed above, except he did not explicitly disclose the concept of "*data input*" as input

controlling of the transformation and translation process. However, Srivastava907 discloses a translation program that includes instrumentation and monitoring by factoring in data input. See figs. 3 and 4. Srivastava907 (col. 6, lines 66 - 67) teaches, "The user supplied UAR 49 are procedures for collecting and analyzing performance data." Thus, it would have been obvious to one skilled in the art, at the time the invention was made, to modify the teachings of Srivastava907, by using data input, to the one or more iterations, and termination, of translation and transformation of platform-specific heterogeneous programs, into Srivastava539, in order to increase optimization, because data inputs can be used to alter the code sequence for the purpose of ultimately producing faster platform-specific executable code.

8. Claims 14 is rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau, U.S. Patent 6,345,311, in view of Srivastava 5,966,539, as applied to claim 13 above, and further in view of common knowledge in computer art.

As per claim 14:

"*...associating a hash value...so that...instructions hash to the hash value.*" Breslau and Srivastava539 do not explicitly address hash values. Official notice is taken that a hash table and associated hash values are well known in the computer art as a common means of indexing into a collection of data and/or code. It would be obvious to one skilled in the art at the time the invention was made to also utilize a hash table feature to position instructions in a collection allowing quick access by an index value, for the purpose of streamlining program organization.

Response to Arguments

9. Applicant has argued, substantially, the following:

(A) Neither of the Srivastava references operates on a heterogeneous program.

The Srivastava539 reference does show several source code modules input into the system (fig. 2), although it does not state that the source modules are heterogeneous. Srivastava does allow for the translation to any one of a heterogeneous output programs. The Breslau reference has been combined with the Srivastava reference to specifically show that the originating environment is heterogeneous.

(B) Claims 19-30 claim functional descriptive material embodied on a computer readable medium, creating functional interrelationships among both the data structure elements and a computing process.

Examiner disagrees with Applicant's interpretation of this issue. The fields of the data structure hold names, not functions. As such these are not functional.

(C) The Srivastava reference ,'539, does not teach "obtaining a binary of a component in a heterogeneous program." The '539 reference doe not recite iterating an intermediate representation of a heterogeneous program.

Srivastava's system does show multiple object code components being input into the system. Breslau's invention expressly states that objects execute in different heterogeneous execution environments. The Srivastava system (fig. 2) does show arrows back to the first block, indicating that a multitude of target computer system programs could be output. See col. 4, lines 53-55, "The specific architectures of target computer systems can be maintained in a CPU architectures description."

(D) Amended claim 15 recites reading in a heterogeneous program having executable components for different architectures and obtaining a binary for a component contained in the heterogeneous program.

The Breslau reference addresses reading in a heterogeneous program, (Abstract, lines 1-7) whereby objects can be instantiated in the other, heterogeneous execution environment.

Conclusion

10. Applicant's arguments with respect to claim 15 have been considered but are moot in view of the new ground(s) of rejection. Applicant's amended claim specified that the component originated in a heterogeneous program. The Breslau reference has been included in the rejection, as it specifically states that objects execute in different, heterogeneous execution environments.

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (703) 305-4564. The examiner can normally be reached Monday through Thursday, from 7:00 A.M. to 5:30 P.M. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Gregory Morse can be reached on (703) 308-4789.

The fax phone numbers are (703) 746-7240 for regular communications and (703) 746-7239 for After Final communications. Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

ANIL KHATRI
PRIMARY EXAMINER

MS

04/18/2003